

relates to grant delay tracking for a particular one of the entities **18**, the memory **34**, which may be a non-volatile memory, stores a maximum delay value that may be transferred into the register **70** at reset or system start.

With the maximum delay value, which may be expressed as time stamp (TS) counts, loaded into the register **70**, processing continues with monitoring for a request event for the entity **18** being monitored (Step **122**). In response to detecting the request event, the time stamp value (TSV) currently on the time stamp bus **84** is latched into the register **72** through the gate **80** (Step **124**). Concurrent, or nearly concurrent with this, the adder **76** is enabled, the results are latched into register **74**, and the comparator **52** is enabled, using a delayed version of the request event signal (Steps **126** and **128**). Thus, the adder **76** adds the request event TSV as captured in the register **72** and the maximum grant delay value as stored in the register **70**, and the sum is transferred to the register **74**, for use by the comparator **52**.

A delay element **78** may be used to generate a delayed version of the request event signal, i.e., REQ', where the non-delayed version is identified as REQ. Generally, the amount of delay time needed will depend on the implementation details, and will be set as a matter of time. It is sufficient to delay long enough to meet the data setup and hold timing requirements for the adder **76** relative to the actual assertion of the REQ signal.

In any case, processing continues with monitoring for the corresponding request grant event (Step **130**). If the request grant event is not detected, processing continues with the comparator **52** comparing the current (dynamically) updating TSV on the time stamp bus **84** with the contents of the register **74** (Step **132**), which contents are the sum of registers **70** and **72**. If the current TSV is greater than the sum held in the register **74**—i.e., the value from adder **76**, the comparator **52** detects that condition as a grant delay violation and asserts an error signal, which can be propagated as needed by the signaling circuits **54** (Steps **134** and **136**). Note that the contents of the contents of registers **72** and **74** may be frozen as part of such processing, for diagnostic use.

On the other hand, if the current TSV does not exceed the sum held in the register **74**, the maximum grant delay is not yet violated and processing returns to monitoring for the request grant event (Step **130**). Thus, if a request grant event occurs before a grant delay violation occurs, processing continues with capturing the TSV on the time stamp bus **84** occurring at the request grant event into the register **74**, thereby overwriting the sum value from adder **76** (Step **138**) and preserving actual request grant time stamp information for diagnostic recording. The request grant signal is also used to disable the comparator and thereby prevent false error assertion (Step **140**).

Those skilled in the art will appreciate that the above process is repeated for subsequent request event detections. Further, as noted, all such processing can be carried out on a concurrent basis for multiple requesting entities **18**.

Thus, even without capturing a running list of arbitration events, the arbitration diagnostic circuit **12** can be configured to provide resource grant delay violation detection and response functions. Broadly, in one or more embodiments, the arbitration diagnostic circuit **12** implements a method of tracking delay times between resource requests and corresponding resource grants for respective ones of two or more entities having arbitrated access to a shared resource. In such embodiments, the arbitration diagnostic circuit **12** detects resource grant delay violations by comparing the delay times to one or more defined delay limits.

Of course, the present invention is not limited by the foregoing discussion, nor is it limited by the accompanying drawings. Indeed, the present invention is limited only by the following claims, and their legal equivalents.

What is claimed is:

1. A method of resource arbitration diagnostic processing comprising:
 - detecting arbitration events for two or more entities having arbitrated access to a shared resource;
 - maintaining a chronological memory trace of the arbitration events; and
 - diagnosing arbitration-related system errors using the chronological memory trace.
2. The method of claim 1, wherein detecting arbitration events comprises detecting resource requests and corresponding resource grants.
3. The method of claim 2, wherein detecting arbitration events further comprises detecting resource releases.
4. The method of claim 1, wherein maintaining a chronological memory trace of the arbitration events comprises maintaining a running list of time-stamped arbitration events.
5. The method of claim 4, wherein maintaining a running list of time-stamped arbitration events comprises storing time-stamped event identifiers for resource request events and resource grant events.
6. The method of claim 5, further comprising storing time-stamped event identifiers for resource release events.
7. The method of claim 5, further comprising detecting excessive resource grant delays based on comparing resource grant delays for given resource request events against corresponding delay limits.
8. The method of claim 7, further comprising calculating the resource grant delays by tracking elapsed times after detecting resource request events, and comparing the elapsed times to maximum grant delay limits defined for the two or more entities.
9. The method of claim 7, further comprising performing one or more of the following actions responsive to detecting an excessive resource grant delay: storing resource grant delay violation information in the running list, freezing the running list, asserting a system halt signal, asserting a delay violation alert signal, capturing arbitration state information, and capturing entity state information for one or more of the two or more entities having arbitrated access to the shared resource.
10. An arbitration diagnostic circuit comprising:
 - an interface circuit configured to detect arbitration events for two or more entities having arbitrated access to a shared resource; and
 - a control circuit configured to maintain a chronological memory trace of the arbitration events.
11. The arbitration diagnostic circuit of claim 10, wherein the arbitration diagnostic circuit is configured to detect resource requests and corresponding resource grants as arbitration events.
12. The arbitration diagnostic circuit of claim 11, wherein the arbitration diagnostic circuit further is configured to detect resource releases as arbitration events.
13. The arbitration diagnostic circuit of claim 12, wherein the interface circuit comprises a signaling interface communicatively coupled to an arbitration circuit, and wherein the interface circuit is configured to receive arbitration event signals from the arbitration circuit.
14. The arbitration diagnostic circuit of claim 10, wherein the control circuit is configured to maintain a chronological memory trace of the arbitration events by maintaining a running list of time-stamped arbitration events.